UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Desarrollo de un sistema empotrado para el manejo remoto de dispositivos informáticos

Sergio Bermúdez Gómez
Tutor: Fernando Jesús López Colino

Junio 2016

Desarrollo de un sistema empotrado para el manejo remoto de dispositivos informáticos

AUTOR: Sergio Bermúdez Gómez
TUTOR: Fernando Jesús López Colino

Hardware & Control Technology Laboratory
Dpto. Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio de 2016

Resumen

Los tiempos que vivimos se caracterizan por el impacto que tienen las tecnologías en todos los ámbitos de nuestras vidas.

Más específicamente, este Trabajo de Fin de Grado engloba las tecnologías de control remoto, las cuales han hecho mucho más sencilla la vida de las personas, ofreciendo toda una serie de posibilidades y eliminando barreras que antes eran insuperables para algunas personas.

No solo han influido en este sentido, también han supuesto la posibilidad de aumentar la productividad y eficiencia en diversos sectores industriales, así como la posibilidad de llegar a conocer sitios y explorar lugares que antes eran inimaginables.

En este Trabajo Fin de Grado nos orientamos en la creación de una plataforma de control remoto que incorpore toda una serie de sistemas de comunicación entre ésta y los diferentes dispositivos que queramos conectar, ofreciendo la posibilidad de ampliar tanto como permita nuestra arquitectura el número y complejidad de los dispositivos.

En concreto, utilizaremos la plataforma Raspberry Pi, sobre la que desarrollaremos un servidor web que permitirá a un usuario la posibilidad de interactuar con los distintos dispositivos conectados a esta.

A su vez, se diseñará una interfaz que integre en un entorno web la interacción del usuario con el entorno del servidor, que sea sencilla, permitiendo el uso de éste a personas sin conocimientos técnicos. Además debe permitir su expansión a más dispositivos sin suponer esto un gran cambio en el proyecto y sin la necesidad de tener conocimientos de programación avanzados. También se proporcionarán toda una serie de librerías que permitan distintos tipos de comunicación entre nuestra plataforma y los dispositivos.

Palabras clave

Control remoto, Raspberry Pi, Servidor Web, Sistema empotrado.

Abstract

The times we live in are characterized by the impact of technologies in all areas of our lives.

In particular, this Final Project Degree focuses on remote control technologies; which have made life much easier for people, offering a range of possibilities and eliminating barriers that were previously insurmountable for some people.

They have not only influenced on this way, they also have offered the possibility of increasing productivity and efficiency in various industrial areas, as well as the possibility to get to know and explore places that were previously unimaginable.

In this Final Project Degree, we focus on creating a remote control platform that incorporates a range of communication systems between it and the different devices to be connected, offering the possibility to expand our architecture to a great number of devices.

Specifically, we will use the Raspberry Pi platform to develop a web server that allows a user to interact with the various devices connected to it.

In turn, an interface that integrates, in a web environment, the user interaction with the server environment, will be designed, which should be simple for the user, allowing the implementation to people without technical knowledge. Also, it must allow its expansion to more devices without assuming a great change in the project; without the need to have advanced programming skills. Finally, a number of libraries that allow different types of communication between our platform and devices will be provided.

Keywords

Remote control, Raspberry Pi, Web Server, Embedded system.

Agradecimientos

En este apartado de mi trabajo de fin de grado me gustaría agradecer a todas aquellas personas que han hecho posible que este proyecto llegué hasta aquí con éxito y que han hecho que su realización sea más amena y sencilla para mí.

En primer lugar me gustaría mencionar a mi tutor, Fernando, el cual contestó a mi petición de realizar un proyecto con él al momento, y el cual me confió este TFG sabiendo que yo no había cursado asignaturas de telecomunicaciones y aun así me dio el apoyo y me animó a embarcarme en éste diciéndome que yo era capaz de hacerlo. Ha estado ayudándome durante todo el curso, sin retrasarse nunca en las contestaciones a mis correos y sin negarse nunca a cederme un poco de su tiempo para resolver mis dudas.

En segundo lugar me gustaría agradecérselo a una compañera de carrera muy especial para mí, Ana, que ha estado a mi lado en los momentos más duros del grado y ha hecho posible que yo esté hoy aquí, escribiendo esto.

En tercer lugar, y no por ello menos importante, dar las gracias a mi familia, que me animaron y me apoyaron económicamente cuando peor estaban las cosas, y gracias a ellos hoy tengo un futuro por delante.

Gracias a todos y cada uno de vosotros.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1 Introducción

1.1 Motivación

La realización de este trabajo viene motivada por el deseo de definir de una plataforma de control remoto propia, de bajo coste y con la posibilidad de controlar todos los aspectos de ésta, para interactuar con una serie de dispositivos electrónicos desde cualquier dispositivo móvil (Tablet, Pc, Teléfono, etc…) a través de un portal web.

El realizar el diseño desde cero nos da la posibilidad de elegir tanto el sistema operativo como las tecnologías implicadas en el control de nuestros dispositivos, pudiendo adaptarla a nuestras necesidades y permitiéndonos expandir nuestra plataforma a otros dispositivos.

Este proyecto está orientado principalmente a controlar dispositivos directamente conectados a nuestra plataforma, en la que alojaremos el servidor web que recibirá las órdenes del cliente conectado a ésta.

Para la realización de este trabajo se identificó previamente una serie de elementos básicos con los que trabajar la plataforma de control remoto, permitiendo así tener una base y unos dispositivos predefinidos que servirán de prototipo para una futura ampliación, manejando los posibles sistemas de comunicación plataforma-dispositivo y sus actuadores.

Para proporcionar la funcionalidad descrita se implementarán una serie de librerías que permitan la comunicación entre la plataforma Raspberry Pi y los dispositivos electrónicos conectados a ella mediante los protocolos de comunicación GPIO, PWM e I2C. Estas librerías constarán de una serie de ficheros a los que llamar especificando los valores de configuración de los distintos modos de comunicación y éstas se encargarán de realizar las configuraciones pertinentes y ejecutar la orden especificada.

El manejo de los dispositivos se realizará mediante una interfaz web. De esta manera se simplificará el uso de éstos por parte del usuario.

1.2 Objetivos

Realizar el diseño e implementación de una plataforma de control remoto de dispositivos electrónicos básicos clasificándolos por el modo de comunicación, que puedan utilizar hacia la plataforma, y que abarcarán toda una posible gama de protocolos para que puedan ser añadidos en el futuro, sirviendo de base a un control remoto con mayor número de plataformas receptoras y mayor número de dispositivos.

En este proyecto se busca implementar una serie de librerías que puedan ser utilizadas por un desarrollador web sin conocimientos técnicos de programación de sistemas empotrados, permitiendo que defina los parámetros necesarios para hacerlos funcionar de una manera genérica.

También se proporcionará una plantilla web que llame a cada uno de los tipos de comunicación implementados para proporcionar un ejemplo de funcionamiento.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Estado del arte**

En este capítulo analizaremos las distintas tecnologías de comunicación remotas de las que disponemos actualmente y haremos una comparativa entre ellas. Después, hablaremos de varios ejemplos de sistemas de control remoto implantados en la sociedad y su funcionamiento, para entender un poco mejor que es lo que se desea implementar.

- **Diseño**

En este capítulo detallaremos el modelo de comunicación web elegido, con una breve explicación, y más adelante se explicaran los tipos de comunicación entre la plataforma y los posibles dispositivos que se conectarán a ella.

También se describirán las tecnologías elegidas para construir las librerías de comunicación con los dispositivos, así como las implicadas en nuestro servidor, comparando siempre sus ventajas y desventajas con el resto de tecnologías disponibles en el mercado.

- **Desarrollo**

En este capítulo se llevará a cabo una descripción de los distintos módulos implementados, describiendo cómo se han codificado y cómo funcionan las distintas librerías y la interfaz de comunicación entre una página web y éstas.

- **Integración, pruebas y resultados**

En este capítulo se detallarán todas las pruebas, tanto unitarias como de integración, llevadas a cabo para comprobar el correcto funcionamiento de nuestros módulos, añadiendo fotos de los resultados obtenidos a modo de pruebas reales.

- **Conclusiones y trabajo futuros**

Por último, se redactarán unas conclusiones sobre el trabajo llevado a cabo durante este proyecto y se plantearán de las posibles líneas de trabajo futuro que quedan abiertas una vez finalizado éste.

2 Estado del arte

Como hemos mencionado anteriormente, la tecnología ha facilitado la vida de las personas hasta llegar un punto en que se puede controlar una gran variedad de dispositivos desde el teléfono móvil u ordenador.

Esto podemos apreciarlo en ejemplos como la domótica, en la que podemos manejar varios dispositivos de nuestra casa a través de una Tablet u otro dispositivo de control remoto; en fábricas, donde se maneja un alto porcentaje de la maquinaria desde un solo ordenador; en museos, donde podemos interactuar con las vitrinas y los objetos expuestos mediante nuestro teléfono móvil.

Todo esto ha sido posible gracias a la informática y, sobre todo, gracias a internet.

2.1 Tecnologías de Control Remoto

2.1.1 Infrarrojos

Desde hace algo más de 30 años se utiliza, con mucha frecuencia, la luz infrarroja para controlar aparatos del hogar a distancia, desde televisores hasta aires acondicionados utilizan este sistema de control remoto.

Este sistema de comunicación es considerado de poco alcance, aunque de forma inalámbrica. Requiere de una corta distancia entre el sistema emisor y el receptor, utilizando una conexión óptica conocida como conexión infrarroja (IRDA).

El funcionamiento de este tipo de control a distancia es muy sencillo y consiste en enviar una señal lumínica desde el aparato de control hasta el aparato receptor de la orden, el cual decodifica la señal y ejecuta el comando recibido (Infrared Data Association, 2011).

Este tipo de comunicación presenta toda una serie de ventajas y desventajas respecto a otras comunicaciones:

Como desventaja podríamos mencionar que este sistema necesita de una trayectoria directa y cercana entre los dispositivos involucrados, y que no permite otra transferencia distinta a la transferencia punto a punto (Santamaría Velázquez, 2015).

Como ventaja podemos destacar que el intercambio de información es rápido y simple, y presenta varios ejemplos en los que es más propicio utilizar este sistema para reducir el coste y la complejidad, por ejemplo, en un sistema de lectura de código de barras de un supermercado.

Esto es así gracias a que ésta tecnología no necesita identificar el objeto que nos proporciona la información, puesto que una conexión entre dos puntos exclusivamente. La tabla 2-1 resume las características de funcionamiento de la tecnología de infrarrojo.

Característica/función	Funcionamiento
Tipo de conexión	Infrarrojos, haz estrecho (ángulo de 30 grados o menos)
Espectro	Óptico, 850 nanómetros (nm)
Potencia de transmisión	100 milivatios (mW)
Velocidad de transferencia de datos	Hasta 16Mbps usando VFIR
Alcance	Hasta 1 metro
Dispositivos soportados	Dos (2)
Canales de voz	Uno (1)
Seguridad de los datos	El corto alcance y estrecho ángulo de haz de infrarrojos ofrecen una forma simple de seguridad
Direccionamiento	Cada dispositivo tiene un identificador físico de 32 bits que se utiliza para establecer una conexión con otro dispositivo.

Tabla 2-1: Características de los infrarrojos

2.1.2 Bluetooth

Surge por la necesidad de comunicación entre los dispositivos móviles y sus accesorios con una interfaz de radio de baja potencia y bajo coste.

Se trata de una red de comunicación Ad-Hoc, es decir, una red inalámbrica descentralizada que no depende de una infraestructura predefinida. Esta tecnología combina conmutación de circuitos y paquetes convirtiéndola en la comunicación idónea para transmitir voz y datos. Utiliza una banda de radio ISM de 2.4 GHz y tiene un alcance de hasta 10 metros de distancia entre dispositivos (Bluetooth SIG, 2016).

Como ventaja respecto a la tecnología infrarroja, podemos hablar de una ampliación de distancia entre los dispositivos involucrados (10 metros actualmente), y la no necesidad de una visión directa entre los dispositivos para el intercambio de información, permitiendo además que puedan conectarse 10 dispositivos diferentes al sistema receptor. Otra ventaja de este sistema de comunicación frente al sistema de infrarrojos es la posibilidad de penetrar objetos sólidos, evitando así la necesidad de tener un canal directo entre los dispositivos.

En cuanto a sus desventajas frente otras tecnologías, podemos hablar de su necesidad de identificar el dispositivo que intenta transmitir datos y el corto alcance que presenta respecto a otras tecnologías que describiremos a continuación. Las características más relevantes se resumen en la tabla 2-2.

Característica/función	Funcionamiento
Tipo de conexión	Expansión de espectro (saltos de frecuencia)
Espectro	Banda ISM de 2.4 GHz
Potencia de transmisión	1 milivatios (mW)
Velocidad de datos total	1 Mbps
Alcance	Hasta 10 metro
Estaciones soportadas	Hasta ocho dispositivos por picorred
Canales de voz	Hasta tres
Seguridad de los datos	Autentificación: clave de 128 bits; cifrado: tamaño de clave configurable, entre 8 y 128 bits
Direccionamiento	Cada dispositivo con una dirección MAC de 48 bits

Tabla 2-2: Características del bluetooth

2.1.3 Redes de Área Local Inalámbricas

Las redes de área local inalámbricas (WLAN) son un sistema de comunicación inalámbrico diseñado para reducir el número de conexiones cableadas, y llevar la información de un punto a otro mediante el uso de ondas de radio evitando así la necesidad de un medio físico guiado.

En cuanto a su funcionamiento, como ya mencionamos, utiliza un sistema de ondas de radio para llevar la información de un punto a otro (Cisco, 2016). El receptor se sitúa en una determinada frecuencia de onda, ignorando el resto de señales y recibe la información almacenándola y transmitiéndola entre la WLAN y la LAN cableada que conecta los dispositivos. El rango de acceso a estos tipos de red puede llegar a unos cien metros o incluso más, y permite la comunicación entre un gran número de dispositivos. La tabla 2-3 resume las características principales de esta tecnología.

Característica/función	Funcionamiento
Espectro	Banda ISM de 2.4 GHz y 5 GHz
Potencia de transmisión	100 milivatios (mW)
Alcance	Hasta 100 metros desde el punto de acceso aproximadamente
Tipos de prestaciones	Múltiples dispositivos por punto de acceso; múltiples puntos de acceso por red
Direccionamiento	Cada dispositivo con una dirección MAC de 48 bits

Tabla 2-3: Características de las redes de área local inalámbricas

2.1.4 Redes Vía Home-RF

Home-RF es un sistema de comunicación inalámbrico que utiliza expansión en frecuencia del espectro de saltos de frecuencia con una velocidad destacable, mejorando a la tecnología bluetooth, y reduciendo la posibilidad de cualquier interferencia entre productos que utilicen ésta misma tecnología en una zona compartida (Lansford, 1999).

Este sistema puede albergar hasta 127 nodos, y es un tipo de comunicación diseñada para su implementación en tarjetas PC-Card. El alcance de transmisión de esta tecnología llega a los 50 metros de distancia entre el dispositivo receptor y el emisor, mejorando así la tecnología bluetooth y la infrarroja. La tabla 2-4 resume las características principales de esta tecnología.

Característica/función	Funcionamiento
Tipo de conexión	Expansión de espectro (saltos de frecuencia)
Espectro	Banda ISM de 2.4 GHz
Potencia de transmisión	100 milivatios (mW)
Velocidad de datos total	1 Mbps o 2 Mbps dependiendo de la modulación de saltos de frecuencia
Alcance	Cobertura media de un hogar
Estaciones soportadas	Hasta 127 dispositivos por red
Canales de voz	Hasta seis
Seguridad de los datos	Algoritmo cifrado de Blowfish
Direccionamiento	Cada dispositivo con una dirección MAC de 48 bits

Tabla 2-4: Características de las redes vía home-RF

2.1.5 Zigbee

Se trata de un conjunto de protocolos de comunicación inalámbrica para radiodifusión digital de bajo consumo. Está basado en las redes WPAN y el objetivo de este sistema de comunicación son las aplicaciones en las que se necesita una comunicación segura con baja tasa de envío y maximización de la vida útil de sus baterías (ZigBee Alliance, 2015).

Esta tecnología utiliza la banda ISM de 2.4 GHz a rango mundial, y la de 868 MHz en Europa, para comunicarse con el resto de dispositivos, y en 16 canales distintos con ancho de banda de 5 MHz. Evita la colisión de transmisiones entre diferentes canales utilizando CSMA/CA (María & Moreno, 2012).

El número de nodos en una misma subred puede ser de hasta 255, pudiéndose formar una red principal de hasta 65535 nodos, y con un alcance de hasta 75 metros.

Como principales ventajas de esta tecnología podemos hablar de su fácil integración, su bajo consumo, su topología de red mallada, su baja tasa de redes de transferencia de datos y su reducción del ciclo de trabajo que proporciona una larga duración de la batería y su bajo coste.

Como desventajas principales podemos destacar su baja tasa de transferencia de datos en comparación con otras tecnologías de comunicación inalámbrica y su menor cobertura al pertenecer al tipo de redes WPAN. La tabla 2-5 resume las características principales de esta tecnología.

Característica/función	Funcionamiento
Espectro	ISM de 2.4 GHz
Alcance	Hasta unos 75 metros

Estaciones soportadas	65535 nodos en la red principal
Canales de voz	16 canales

Tabla 2-5: Características de Zigbee

2.2 Ejemplos de Sistemas de Control Remoto

Actualmente existen toda una serie de sistemas de control remoto utilizando las diferentes tecnologías descritas, ejemplos que nos sirven como base de nuestro proyecto y como modelos a seguir para la realización de este, actuando como fuente de inspiración para desarrollar nuestra propia plataforma de control remoto.

2.2.1 Domótica

Un ejemplo de cómo ha mejorado la vida de las personas y de las posibilidades que nos ofrece el control remoto de dispositivos es la domótica.

Conocemos como domótica al conjunto de tecnologías involucradas en el manejo y automatización de una casa, permitiendo gestionar los distintos dispositivos electrónicos de ésta, y de los dispositivos que ahora incorporan un sistema de recepción remota para pasar a ser dispositivos controlados digitalmente (CEDOM, 2015).

Un sistema domótico tiene la capacidad de recoger información proveniente de unos sensores de entrada, interpretar y procesar dicha información y construir una respuesta a modo de orden mediante unos actuadores.

También tiene la posibilidad de acceder a las redes de comunicación que interactúen con la casa.

La domótica ha contribuido en diferentes aspectos a la hora de manejar el entorno de una casa, ayudando a reducir el coste energético (gestionando inteligentemente iluminación, climatización, agua caliente, el uso de electrodomésticos.), aumentando la accesibilidad recortando las barreras de distancia que puedan sufrir algunas personas, proporcionando una mayor seguridad mediante un sistema de vigilancia controladas por un sistema remoto, etc... Todo esto se debe a la aparición de los sistemas de control remoto, actualmente utilizando el sistema de red de una vivienda, conectando nuestros dispositivos y nuestro sistema de control a internet.

2.2.2 Mercedes Me

Otro ejemplo de control remoto implantado en nuestra sociedad es el que incorporan los coches de última generación, como es el caso de la marca Mercedes, la cual incorpora el sistema Mercedes Me, la cual permite controlar varios aspectos del vehículo desde una aplicación móvil a través de la conexión de internet que presenta.

El vehículo incorpora un módulo de comunicación denominado Combox, el cual envía la información del vehículo a un servidor seguro al cual el dueño puede acceder a través de internet (Mercedes-Benz, 2016).

Activar la calefacción, mostrar el navegador, la posición del vehículo o cerrar y abrir las puertas son algunos ejemplos de las diferentes posibilidades existentes (Arnal, 2015).

2.2.3 Comunicación Remota Industrial

En el sector de la industria tenemos presente una gran diversidad de accesos remotos a diferentes sistemas de control y manejo de las distintas partes de la producción (Siemens AG, 2016).

Gracias a la red, los operarios pueden comunicarse con las distintas máquinas o los distintos sistemas de control, para verificar el estado de éstos e incluso para cambiar diferentes valores en la forma de operar de éstas.

Todo esto ha permitido la expansión global de la producción, pudiendo localizar fábricas en puntos del mundo sin necesidad de estar presentes físicamente en ellos para controlar distintas tareas, o evitar riesgos a las personas que trabajan en ellas proporcionando la posibilidad de comprobar valores de estado sin acercarse a un lugar de riesgo.

3 Diseño

En este apartado mostraremos el diseño de nuestra plataforma de control remoto, explicando cual será el modelo de comunicación vía web y los distintos diseños de comunicación entre nuestra plataforma hardware y los dispositivos electrónicos a controlar.

3.1 Hardware

En nuestro proyecto utilizaremos la plataforma Raspberry Pi 2 modelo B (figura 3-1) como sistema de control de los dispositivos, alojando en ésta nuestro servidor web. Como sistema operativo de nuestra plataforma hemos escogido Raspbian, el cual describiremos en la sección 3.2.2.

3.1.1 Raspberry Pi 2 Model B

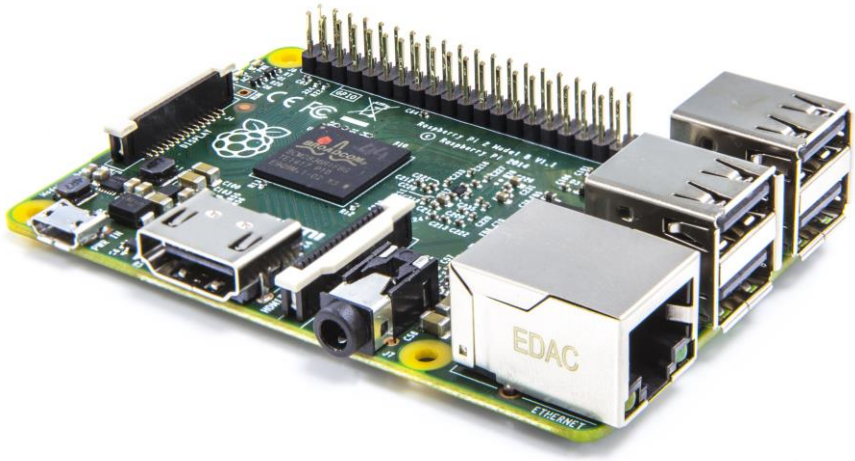

Figura 3-1: Rasberry Pi 2 Model B

Raspberry Pi es un ordenador de placa reducida, es decir, de placa única, de bajo coste, que fue desarrollada en Reino Unido por la propia Fundación Raspberry Pi (Raspberry PI Foundation, 2012b).

Fue diseñada para fomentar la enseñanza de ciencias de la computación en los centros educativos.

Podemos considerarlo un producto con propiedad registrada pero de uso libre, puesto que disponen de contratos de distribución y venta con dos empresas y al mismo tiempo permiten que cualquier persona pueda comerciar libremente con este hardware. La plataforma mantiene así el control, pero ofreciendo la posibilidad de su uso libre en el ámbito educativo y particular.

El software de Raspberry Pi es open source, cuya versión oficial de sistema operativo describiremos más adelante (Raspbian), aunque permite una gran variedad de sistemas operativos.

Otras características técnicas de esta plataforma son:

- Diseño que incluye un System-on-a-chip Broadcom BCM2835, que contiene un procesador central ARM1176JZF-S a 700 MHz
- Posibilidad de hacerle overclock de hasta 1 GHz sin perder la garantía.
- Procesador gráfico GPU VideoCore IV
- 1 Gb de memoria RAM
- Uso de tarjeta SD como sistema de almacenamiento

Como ventajas frente a otras plataformas podemos destacar (Gus, 2015):

- Gran capacidad de la memoria RAM (1 Gb por los 512 Mb de la Beaglebone Black) y procesador Quad core, lo que permite una mayor velocidad y rendimiento a la hora de ejecutar paquetes software en ella
- Puerto HDMI de tamaño normal, lo que permite conectar la placa directamente a un monitor y trabajar sobre ella si no disponemos de conexión ssh
- 4 puertos USB frente a 1 solo puerto de la Beaglebone Black
- Sistema operativo Raspbian, específicamente diseñado para ésta y producido por Debian

3.2 Software

Respecto al software de nuestro sistema cliente-servidor, implementamos un sistema de comunicación HTML – PHP, en la que el código HTML se mostrará en la parte cliente y el código PHP se evalúa en la parte servidor. La figura 3-2 muestra el diagrama de comunicación propuesta.

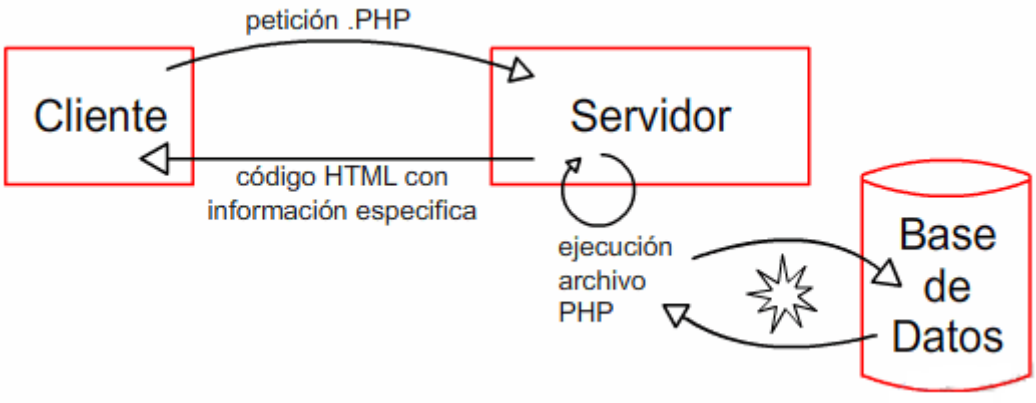

Figura 3-2: Diseño de comunicación web

Dentro del código PHP implementaremos llamadas a código Python, que será el encargado de realizar la comunicación plataforma – dispositivo. En caso de necesitar el uso de bases de datos en un futuro, será la parte del servidor la encargada de comunicarse con ésta.

3.2.1 Modelo Cliente - Servidor

Nuestro proyecto de control remoto implementará el modelo cliente – servidor web como el mostrado en la figura 3-3.

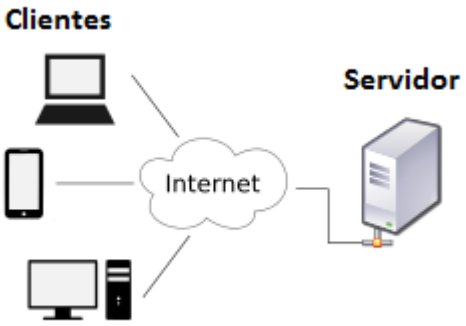

Figura 3-3: Modelo cliente-servidor

Nuestro cliente constará de una sencilla vista que proporcionara un formulario para el envío de órdenes al usuario para introducir una serie de valores y enviarlos o recogerlos mediante botones, y mandará dicha información a nuestro servidor web a través de la red de internet como puede verse en la figura 3-3.

El servidor interpretará qué botón ha sido pulsado, recogerá los valores, y ejecutara las llamadas necesarias a nuestros ficheros Python para activar y manejar nuestros dispositivos conectados.

3.2.2 Raspbian

Se trata de un sistema operativo distribuido por GNU/Linux, lanzado al público en Junio de 2012 y basado en Debian Wheezy, diseñado específicamente para Rasberry Pi (Debian, 2009).

Como características principales tiene:

- Distribución LXDE como escritorio
- Midori como navegador web
- Herramientas de desarrollo como IDLE
- Menú raspi-config para cambiar la configuración del sistema operativo sin tener que acceder a archivos manualmente

Las ventajas de este sistema operativo frente a otros son:

- Proporciona soporte optimizado para operación en coma flotante mediante hardware, lo que ofrece un mayor rendimiento
- Incorpora la herramienta IDLE para el uso de lenguaje Python, el cual hemos elegido para nuestro proyecto y del que se hablará más adelante
- Al ser una distribución de Linux, tanto la terminal como el sistema de archivos utilizan la misma interfaz de navegación y los mismos comandos, sin necesidad de aprender nuevos para su manejo
- Se trata de una distribución de código abierto, lo que nos permite una mayor libertad a la hora de realizar cambios en el sistema operativo si fueran pertinentes
- Otra ventaja de utilizar una distribución de Linux para nuestra Raspberry es que asemeja la plataforma a un ordenador personal, permitiendo un mayor enfoque de ésta que el que tendría como solo una placa con microcontrolador

3.2.3 Servidor Web HTTP Apache

Nuestra Raspberry integra un servidor apache para alojar la parte del servidor web.

Es un servidor web HTTP de código abierto, diseñado para muchas plataformas, incluida nuestra plataforma GNU/Linux. Su nombre proviene de la tribu de los Apaches, últimos en rendirse ante sus conquistadores, puesto que alguien en el entorno de desarrollo de este servidor quería relacionarlo con los conceptos de firme y enérgico, pero no agresivo. Este servidor lo desarrollan y mantienen una comunidad de usuarios bajo la tutela de Apache Software Foundation dentro del proyecto HTTP Server (httpd) (The Apache Software Foundation, 2007).

Tuvo un gran impacto en el desarrollo de la World Wide Web y llegó a su máximo esplendor en 2005 logrando un 70% de presencia en los sitios web a nivel mundial.

Las características principales de esta tecnología son:

- Implementa el protocolo HTTP
- Bases de datos de autenticación y negociado de contenido
- Falta de interfaz gráfica
- Es un sistema Multi-plataforma
- Arquitectura modular

Las ventajas de esta tecnología frente a otras son:

- Es el servidor más usado desde 1996, por lo que podemos encontrar multitud de documentos e información sobre éste, así como ayuda y soporte.
- Es tecnología de código abierto, proporcionando mayor libertad a la hora de desarrollar y realizar modificaciones.
- Al ser un servidor con arquitectura modular podemos localizar las distintas funcionalidades de este con mayor facilidad y realizar las modificaciones u obtener la información deseada más cómodamente.

3.2.4 HTML

Implementa la vista de la página en el cliente, y consta de un formulario para enviar e interactuar con los diferentes puertos de la Raspberry.

HTML (HiperText Markup Language) surgió para satisfacer las necesidades de la Web, la cual tenía que ser distribuida, debía ser hipertexto y fácilmente navegable a través de ella, tenía que ser compatible con todo tipo de ordenadores y sistemas operativos, y debía ser dinámica, con un proceso de intercambio y actualización rápido y ágil (W3C, 2014).

Así surge este lenguaje de programación web, el cual convierte un archivo de texto en una página web con la capacidad de mostrar imágenes, animación, formularios, etc...

Las ventajas o motivaciones para utilizar este lenguaje son:

- Disponemos de toda una serie de editores ya contrastados para trabajar este lenguaje y aprovechar todas sus posibilidades
- Es un lenguaje que trabajamos en la carrera y tenemos una base sobre la que apoyarnos
- Para realizar pequeños cambios puede utilizarse un simple editor de textos y evitamos así dependencias con herramientas concretas de creación de páginas web.

- Es un lenguaje muy intuitivo y descriptivo, fácil de mantener y trabajar
- Es soportado por todos los navegadores actuales

3.2.5 PHP

PHP es un lenguaje script interpretado en el lado del servidor utilizado en la creación de páginas Web dinámicas y contenidas en páginas HTML. Su sintaxis proviene de C, Java y Perl añadiendo alguna característica propia (The PHP Group, 2008).

Al ejecutarse en el lado del servidor nos permite acceder a bases de datos o contenidos que tenga el propio servidor en sus directorios internos, y al ser ejecutado en éste, no necesita que el navegador del cliente lo soporte, creando así una independencia de entornos entre el cliente y el servidor.(Figueroa, 2016)

En nuestro proyecto, interpreta las peticiones del cliente, recogiendo por el método REQUEST los distintos valores en función de que botón ha sido accionado, y ejecutando por línea de comando el fichero Python específico.

Como ventajas o motivaciones para utilizar este lenguaje tenemos (Brezo, 2015):

- Soporta una gran cantidad de tipos de bases de datos que podamos necesitar en proyectos futuros
- Integra varias bibliotecas externas que nos proporcionan toda una serie de funcionalidades ya implementadas.
- Es un lenguaje que se trabaja en la carrera y tenemos una base sobre la que apoyarnos
- Ofrece una solución sencilla y universal, puesto que es el lenguaje del lado del servidor más conocido (24% de las páginas existentes lo incorporan)
- Es un lenguaje libre y abierto

3.2.6 Python

Python es un lenguaje de programación interpretado que se rige por la máxima de construir un código totalmente legible y fácil de seguir.

Como características de este lenguaje (Python Software Foundation, 2016a):

- Es un lenguaje multiparadigma, es decir, que permite tanto programación orientada a objetos como programación imperativa o programación funcional
- Es un lenguaje interpretado que utiliza tipado dinámico
- Es un lenguaje multiplataforma
- Su licencia es de código abierto, y recibe el nombre de Python Software Foundation License, compatible con la licencia de GNU 2.1.1

Las ventajas de utilizar este lenguaje frente a otros son (Peralta, 2012):

- Como ya hemos mencionado, dispone de una gran cantidad de librerías ya implementadas para el manejo de puertos de una Raspberry Pi
- Permite una mayor libertad de programación al no tener que declarar el tipo de las variables
- El código Python es sencillo y legible, con pocas líneas de código se logra el resultado deseado

- Es un lenguaje portable, admitido por todos los sistemas operativos
- Es un lenguaje muy trabajado actualmente y tenemos una gran cantidad de información y documentación disponible

3.3 Comunicación con dispositivos externos

Se ha planteado la comunicación con los dispositivos electrónicos utilizando tres aproximaciones GPIO, PWM e I2C, las cuales abarcan la mayoría de dispositivos electrónicos dependientes de éstas para su comunicación.

3.3.1 Comunicación mediante GPIO

La comunicación por GPIO es la que se vale de un pin genérico cuyo comportamiento se puede controlar o codificar por el usuario en tiempo de ejecución. A este pin se le denomina GPIO, y puede ser tanto de entrada como de salida.

Estos pines no tienen ningún objetivo predefinido y tampoco se les asigna una función de forma predeterminada, si no que se proporcionan al usuario para utilizarlos de forma libre para la funcionalidad que necesite, siempre y cuando cumpla con el uso de la señal digital que estos manejan (Wikipedia, 2015).

En cuanto al uso de este tipo de pines, suelen estar presentes en chips con pocos pines, chips multifunción y aplicaciones embebidas como es el caso de la Raspberry Pi que hacen un uso intensivo de éstos para la lectura de sensores o para enviar señales de activación a diferentes periféricos.

Para este tipo de comunicación será necesario configurar y definir qué pin vamos a utilizar, es decir, especificar el número de pin de la Raspberry; qué sentido tomará esta señal, es decir, si será una señal de salida o por el contrario el puerto recibirá una señal de entrada; y qué valor toma este pin, siendo éste un valor lógico (Raspberry PI Foundation, 2012a).

3.3.2 Comunicación mediante PWM

Este protocolo de comunicación está basado en la modulación por ancho de pulsos (definición de la cuál proceden sus siglas) de una señal o fuente de energía. Se trata de una tecnología que modifica el ciclo de trabajo de una señal periódica para transmitir información utilizando un canal de comunicaciones o para controlar la cantidad de energía que se envía a una carga (Wikipedia, 2016).

La definición de ciclo de trabajo de esta señal periódica es el ancho relativo de la parte positiva en relación con el período:

$$D = \frac{\tau}{T}$$

Donde D es el ciclo de trabajo, t es el tiempo en que la función es positiva (ancho de pulso) y T el perído de la función.

En cuanto al uso de este protocolo de comunicación, está presente en muchos circuitos integrados para lograr circuitos funcionales que puedan controlar fuentes conmutadas, controles de motores, controles de elementos termoeléctricos y choppers para sensores en ambientes ruidosos, entre otras funcionalidades.

La compañía más importante que fabrica este tipo de integrados es Texas Instruments.

Un ejemplo de circuito PWM queda descrito en la figura 3-4.

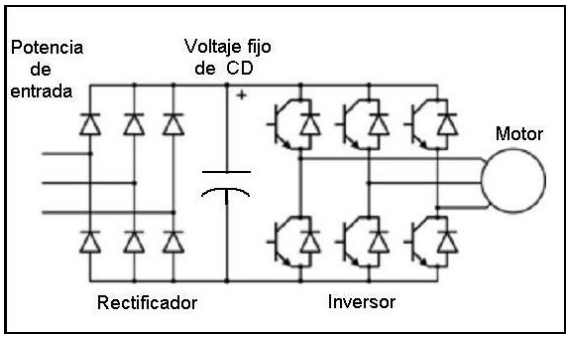

Figura 3-4: Circuito modulador de motor por PWM

Para este tipo de comunicación será necesario configurar y definir qué pin vamos a utilizar, es decir, especificar el número de pin de la Raspberry; la frecuencia en Hertzios a la que funcionará el PWM, y el ciclo de trabajo deseado para éste.

3.3.3 Comunicación mediante I2C

La comunicación I2C fue diseñada por Philips a mediados de los años ochenta, para conseguir una fácil comunicación entre componentes pertenecientes a un mismo circuito.

La velocidad original de comunicación fue definida en un máximo de 100 kbit por segundo en sus comienzos, y actualmente se ha logrado una velocidad de transmisión de hasta 5 Mbits por segundo con el modo UFM.

La comunicación I2C está diseñada como un bus maestro-esclavo. La transferencia de datos la inicia siempre un maestro y el esclavo reacciona a esta petición. Existen otras posibilidades como la comunicación multimaestro en que éstos pueden comunicarse entre ellos convirtiéndose en maestros y esclavos al mismo tiempo. El control de acceso al bus se define por las especificaciones permitiendo que los maestros puedan ir turnándose (I2C-Bus, 2016).

Un dispositivo preparado para la comunicación I2C dispone de dos líneas de señal:

- SCL, que constituye la línea de la señal de reloj, y cuya señal siempre es generada por el maestro
- SDA, que constituye la línea de transferencia de datos

Ambas líneas necesitan resistencias pull-up en su conexión con la línea de alimentación.

En cuanto al uso de este protocolo de comunicación, está presente en aquellos sistemas en los que se necesita que un microcontrolador maneje toda una red de circuitos integrados, ya que se aprovecha la sencillez que éste presenta al requerir solo dos pines para llevar a cabo dicho control y al software tan sencillo que éste necesita para llevarse a cabo, factores que influyen en una reducción de costes en desarrollo, producción y pruebas.

Actualmente existen modos de comunicación más rápidos, pero I2C, debido a su bajo coste, es idóneo para sistemas periféricos en los que no prima la velocidad. Un diagrama de ejemplo de comunicación maestro-esclavo mediante I2C se muestra en la figura 3-5.

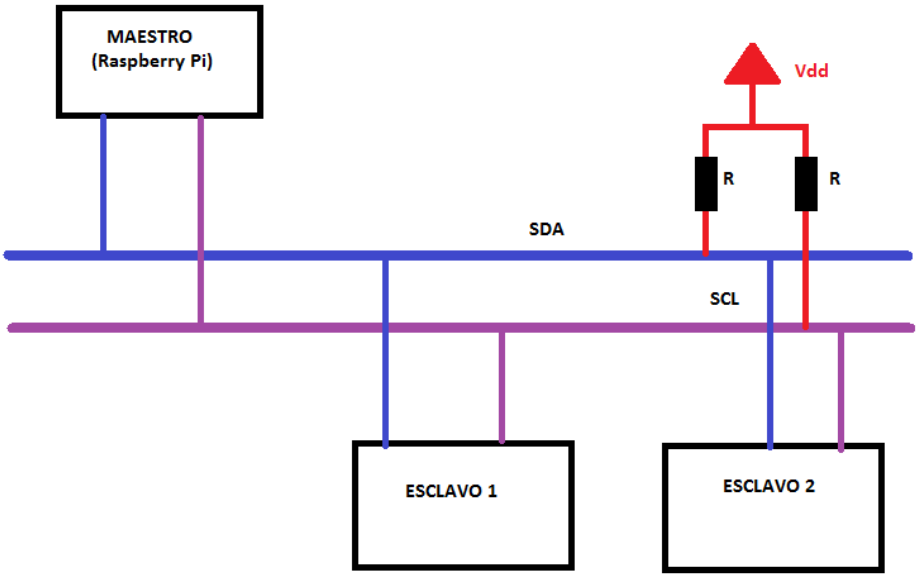

Figura 3-5: Modelo maestro-esclavo

Para este tipo de comunicación será necesario especificar la dirección (en el bus de comunicaciones) del dispositivo conectado a la plataforma, la dirección del registro sobre el cual queremos trabajar, y en caso de ser una operación de escritura, el valor que queremos enviar a éste.

4 Desarrollo

En este apartado se detallará la implementación en cuanto al código implicado en la comunicación de nuestra plataforma con los dispositivos y en cuanto al código que realiza la comunicación del cliente con el servidor web.

4.1 Comunicación Cliente – Servidor

4.1.1 Index.php

Se trata de un fichero php que contiene tanto la vista html de nuestra interfaz de comunicación entre el usuario y nuestra placa, como el php que interpreta las órdenes dadas por éste en nuestro servidor.

En cuanto al html, se trata de la implementación de una serie de botones y cuadros de texto que permiten interactuar al usuario con nuestros periféricos. Proporcionando la vista mostrada en la figura C-5 del anexo.

En cuanto al php, interpreta cada pulsación de un botón en la vista y llama al fichero Python específico para llevar a cabo cada acción deseada por el usuario.

Las llamadas a Python se realizan mediante el comando exec, que simplemente ejecuta un fichero Python utilizando la terminal de la Raspberry y pasándole por argumento a cada fichero los valores introducidos en los cuadros de texto por nuestro usuario.

4.2 Módulo GPIO

4.2.1 Hardware

En la parte de desarrollo hardware, conectamos un dispositivo que utilice comunicación mediante GPIO a nuestra plataforma Raspberry, utilizando los puertos GPIO de los que esta dispone, y diseñando un circuito electrónico adecuado para este dispositivo.

Un ejemplo de circuito sería el descrito por la figura C-1 en el anexo.

4.2.2 Software

Se trata de un conjunto de ficheros con código Pyhton que manejan la comunicación con los dispositivos que utilizan un puerto GPIO.

Cada uno de estos ficheros importa la librería RPi.GPIO, librería de distribución libre y que nos será muy útil para acceder a las funcionalidades que nos ofrecen este tipo de puertos de la Raspberry (Python Software Foundation, 2016b).

4.2.2.1 *EnciendeGpio.py*

Se trata de un fichero Python que recibe un entero por parámetro de entrada que interpreta como el puerto GPIO sobre el que trabajar, y que comienza estableciendo el sistema de numeración BCM para localizar éste.

El sistema de numeración BCM hace referencia a la forma de referenciar los pines de nuestra placa, en este caso, seguirían el siguiente dibujo:

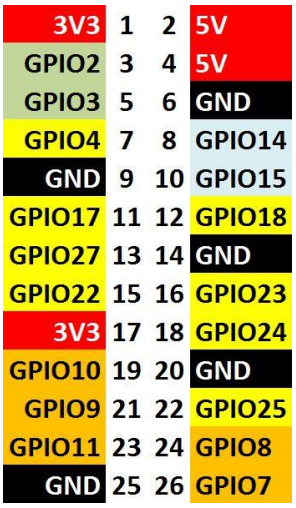

Figura 4-1: Pines Raspberry con numeración BCM

En la figura 4-1 podemos observar cual sería la numeración BOARD (numeración interior) y la numeración BCM en la que indicaríamos el número de GPIO y no el número de pin (numeración exterior).

Una vez identificado el número de GPIO al que tenemos conectado nuestro dispositivo, lo inicializa a modo OUT para que obtenga un funcionamiento de salida, utilizando una llamada a la función setup (número gpio, dirección).

Por último, activa el puerto de salida enviándole un '1' y valiéndose de la función output (número gpio, valor).

4.2.2.2 *ApagaGpio.py*

Este fichero Python es muy similar al descrito en el apartado anterior solo que de opuesto funcionamiento.

En primer lugar estable el sistema de numeración BCM, seguido de establecer la dirección de nuestro puerto que una vez más recibimos por los argumentos de la ejecución a modo de número entero.

Por último, establece el valor de salida de nuestro puerto pasándole un '0' como valor a la función output (número gpio, valor) y termina utilizando la función cleanup() que liberará el puerto GPIO para poder usarlo de distinto modo en futuras ocasiones.

4.2.2.3 *ParpadeaGpio.py*

Este fichero Python agrupa las dos funcionalidades anteriores agregándole un factor tiempo.

En primer lugar importa la librería time de Python, necesaria para una futura llamada a las funciones de espera activa que necesitaremos para llevar a cabo la funcionalidad.

Establece el modo de numeración a BCM y la dirección del puerto GPIO que recibimos por línea de parámetros de la ejecución, y una vez hecho esto, entra en un bucle en el que hará parpadear el LED hasta N veces, siendo N un valor indicado como segundo argumento por parámetros de ejecución.

Para llevar a cabo el parpadeo, ira poniendo a '1' o '0' el valor del GPIO intercalándolo estas llamadas a la función output con una llamada a la función sleep (tiempo en segundos)

indicándole el tiempo en segundos obtenido por parámetros de llamada al programa como número entero.

Finalmente liberará el GPIO con una llamada a la función cleanup ().

4.3 Módulo PWM

Este módulo implementa la comunicación mediante PWM con los posibles dispositivos electrónicos conectados a ésta.

4.3.1 Hardware

En la parte de desarrollo hardware, conectamos un dispositivo que utilice comunicación mediante PWM a nuestra plataforma Raspberry, utilizando los pines de posible configuración PWM de los que ésta dispone de los que esta dispone, y diseñando un circuito electrónico adecuado para este dispositivo.

Un ejemplo de circuito está descrito por la figura C-1 en el anexo.

4.3.2 Software

En la parte de desarrollo software, utilizamos la tecnología Pyhton anteriormente descrita para encender y apagar el LED, enviando una señal de activación a nuestro puerto o una señal de desactivación según la acción que necesitemos, con unos valores específicos de frecuencia y ciclo de trabajo. La llamada a Python se ejecuta desde la región de código que se ejecuta al pulsar un botón en nuestra página web, tras haber introducido el puerto que hemos conectado al pwm, el valor de frecuencia y el valor de ciclo de trabajo.

El módulo PWM consta de los siguientes ficheros:

4.3.2.1 *IniciarPWM.py*

Este fichero Python recibe por argumentos 4 valores:

- Un entero que indicara el puerto PWM conectado a nuestro dispositivo
- Un entero que representará la frecuencia en Hertzios del PWM
- Un entero entre 0 y 100 que representará el ciclo de trabajo del PWM
- Un entero que representará el tiempo en segundos de funcionamiento del PWM

Nuestro fichero, una vez convertido todos estos valores a un entero interpretable por Pyhton mediante una conversión, utilizará el primer valor para poner a modo OUT la dirección de este puerto mediante la función setup (), después utilizara la función GPIO.PWM (número gpio, frecuencia) para configurar el puerto GPIO a modo PWM, y guardar el valor de retorno en una variable para más tarde trabajar sobre ella.

Con una llamada a la función start (ciclo de trabajo) sobre el valor obtenido en la llamada anterior, haremos funcionar el PWM, y esperaremos el tiempo indicado en segundos para apagarlo posteriormente mediante una llamada a la función stop ().

4.3.2.2 *PararPWM.py*

Este fichero Python recibe por argumentos el número del GPIO configurado como PWM y activo en este momento.

Obtiene el valor de PWM de este puerto GPIO mediante el uso de la función GPIO.PWM de Python, y realiza una llamada a la función stop () para pararlo.

4.3.2.3 *CambiarValoresPWM.py*

Este fichero recibe por argumentos los siguientes valores:

- Un entero que indica el número de GPIO configurado como PWM
- Un entero que indica la nueva frecuencia en Hz que queremos establecer
- Un entero entre 0 y 100 que indique el nuevo ciclo de trabajo a establecer en el PWM activo

El fichero recoge y transforma estos valores de entrada a tipos interpretables por Python, y seguido obtiene el valor PWM mediante la función GPIO.PWM (). Una vez identificado, utiliza la función ChangeFrequency (valor) y ChangeDutyCycle (valor) para cambiar los valores del PWM.

4.4 Módulo I2C

Este módulo implementa la comunicación I2C de nuestra Raspberry con los dispositivos conectados mediante este protocolo.

4.4.1 Hardware

En la parte de desarrollo hardware realizamos la conexión de dos dispositivos diferentes que utilizan comunicación I2C para comprobar el correcto funcionamiento de este módulo.

Como primer dispositivo conectamos una placa expansora de I2C que consta de unos LEDs que se encienden representando el valor binario que se le envía a un específico registro.

El ejemplo de circuito electrónico entre la plataforma y la placa expansora puede verse en la figura C-3 del anexo.

Como segundo dispositivo elegimos el motor de un disco duro antiguo proporcionado por la universidad, para el cual realizamos un estudio de sus características y las posibilidades que este ofrecía.

Como driver para hacerlo funcionar escogemos el DRV10983 de Texas Instruments. Se trata de un driver para motores de 3 fases, de 24 voltios y comunicación I2C, el cual nos permite conectarlo al conector de 11 pines de nuestro disco duro.

Para ello, diseñamos el circuito cuyo esquemático se muestra en la figura 4-2.

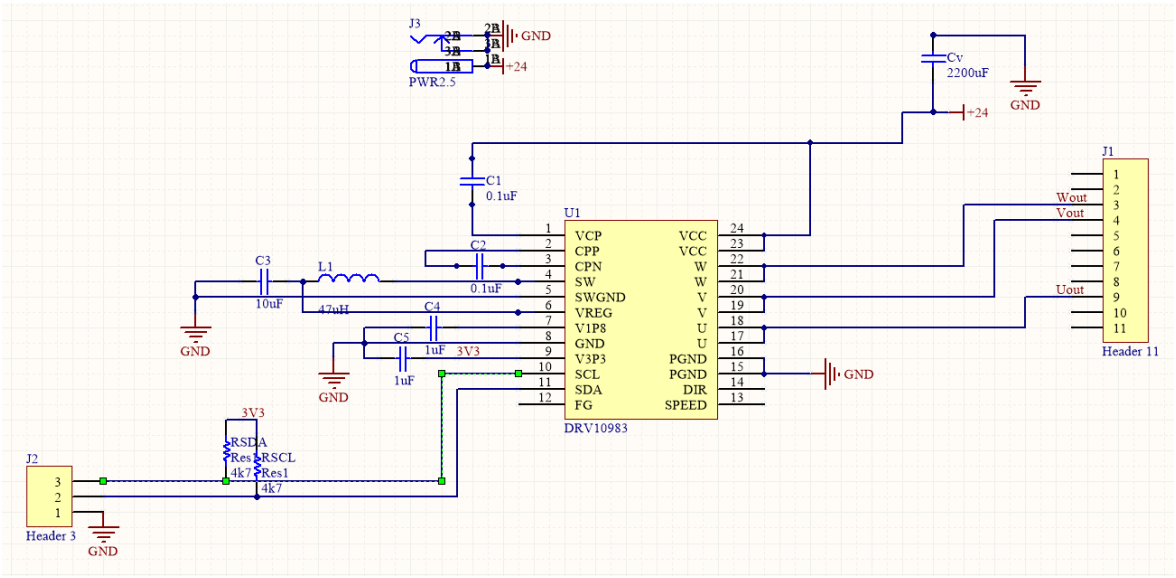

Figura 4-2: Circuito esquemático Raspberry-motor disco duro

En el que se aprecia un conector de tres pines para la Raspberry, coincidiendo con las líneas SDA, SCL y GND, conectadas a las entradas correspondientes en el driver, y éste a su vez, conecta las salidas U, V y W al conector del disco duro.

También proporcionamos al driver un Jack de alimentación de 24 Voltios, y conectamos el resto de salidas a un circuito de condensadores, resistencias y transistores sugerido por el fabricante. Una vez hecho esto, generamos el PCB (figura 4-3).

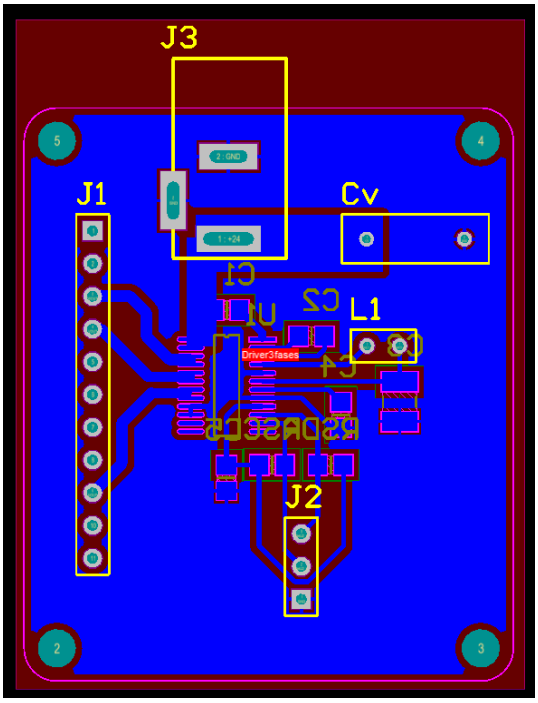

Figura 4-3: PCB Raspberry-motor disco duro

4.4.2 Software

La parte de desarrollo software consta de una serie de ficheros Python que implementan la funcionalidad de enviar datos y leerlos de un dispositivos especificando su dirección de localización y registro.

Estos ficheros importarán la librería SMBUS de Python, la cual implementa un subconjunto de protocolos I2C, presentes en todas las placas de última generación y que se utiliza sobre todo para obtener información del estado del hardware de la plataforma (Azzam, 2016).

Este subconjunto funciona de acuerdo al sistema de comunicación maestro-esclavo, y en este caso lo utilizaremos para obtener el estado de las líneas SDA y SCL en el momento de la ejecución.

4.4.2.1 *EscribirI2C.py*

Se trata de un fichero Python que recibe los siguientes valores por línea de parámetros:

- Un hexadecimal de 8 bits escrito en forma de dos cifras (entre 00 y FF) que indicará la dirección de nuestro dispositivo I2C conectado
- Un hexadecimal de 8 bits escrito en forma de dos cifras (entre 00 y FF) que indicará la dirección del registro al que queremos enviar un valor
- Un hexadecimal de 8 bits escrito en forma de dos cifras (entre 00 y FF) que será el valor a enviar a nuestro registro específico

En primer lugar, el fichero lleva a cabo una serie de conversiones entre los valores recogidos y valores enteros que pueda interpretar la librería de comunicación I2C.

Después, llevará a cabo una llamada a la función write_byte_data (direccion_dispositivo, direccion_registro, valor_entero) que enviará el valor a nuestro dispositivo.

4.4.2.2 *LeerI2C.py*

Se trata de un fichero Python que recibe los siguientes valores por línea de parámetros:

- Un hexadecimal de 8 bits escrito en forma de dos cifras (entre 00 y FF) que indicará la dirección de nuestro dispositivo I2C conectado
- Un hexadecimal de 8 bits escrito en forma de dos cifras (entre 00 y FF) que indicará la dirección del registro del que queremos obtener un valor

El fichero realiza una serie de transformaciones necesarias entre los valores recibidos y los valores interpretables por Python y después realizará una llamada a la función read_byte_data (direccion_dispositivo, direccion_registro), la cual retorna un valor que será recogido y mostrado por pantalla.

5 Integración, pruebas y resultados

En este apartado de la memoria se describirá la batería de pruebas llevada a cabo para comprobar la correcta implementación de las librerías de manejo de GPIO, I2C y PWM, así como la interfaz web creada para el control remoto de la plataforma Raspberry Pi.

La batería de pruebas se divide en dos tipos:

- Pruebas Unitarias, que comprobarán uno a uno cada uno de los ficheros python implementados realizando ejecuciones por línea de comandos y analizando los resultados obtenidos con los esperados para determinar si cumple con la funcionalidad deseada.
- Pruebas de Integración, que probará la correcta implementación de la página web y la capacidad de llamar desde ésta a cualquiera de los módulos implementados, añadiendo uno a uno cada módulo a esta y repitiendo las pruebas unitarias con todo el proyecto embebido en la página web.

5.1 Pruebas Unitarias

En esta parte de las pruebas, se ejecutarán uno a uno cada uno de los ficheros por la terminal de la Raspberry, utilizando permisos de súper usuario añadiendo el comando sudo a cada una de las llamadas, puesto que los ficheros necesitan este permiso para escribir en los registros de la Raspberry adecuados para el modo de comunicación. Se compararán los resultados obtenidos con los resultados esperados para cada una de las pruebas asegurando así su correcta implementación.

5.1.1 Módulo GPIO

Probamos a encender un LED, conectado a nuestra plataforma siguiente el diseño de circuito descrito en la figura C-1 del anexo C, utilizando el fichero Python llamándolo por nuestra terminal:

$ sudo python /var/www/leds/gpio/enciende.py '22'

El LED se enciende correctamente.

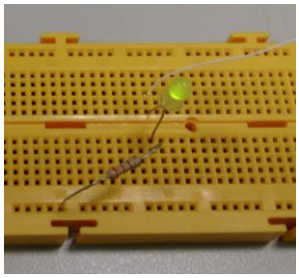

Figura 5-1: LED encendido

A continuación, probamos a apagar el LED mediante el uso de nuestro fichero Python de apagado del LED:

$ sudo python /var/www/leds/gpio/apaga.py '22'

El LED se apaga correctamente.

Por último, ejecutamos una llamada al fichero de parpadeo de nuestro LED:

$ sudo python /var/www/leds/gpio/parpadea.py '22' '5'

El LED parpadea durante 5 segundos.

5.1.2 Módulo PWM

Conectamos el puerto PWM a una sonda de un osciloscopio para comprobar que señal produce, y ejecutamos una llamada al fichero Python iniciarPWM.py por terminal, enviando 100 Hz como frecuencia y 50 % como ciclo de trabajo:

$ sudo python /var/www/pwm/iniciarPWM.py '11' '100' '50'

En la pantalla del osciloscopio comprobamos que la señal del PWM es la siguiente:

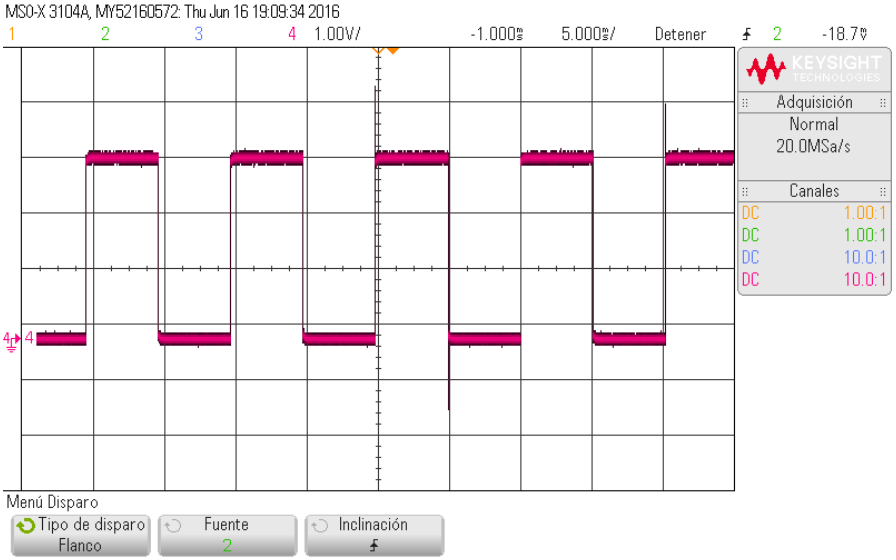

Figura 5-2: Señal PWM 1

En la figura 5-1 observamos que la señal tiene la forma esperada.

Realizamos una serie de pruebas con el fichero cambiarValoresPWM.py para comprobar que cambian estos en el parpadeo del LED:

$ sudo python /var/www/pwm/cambiarPWM.py '11' '80' '20'

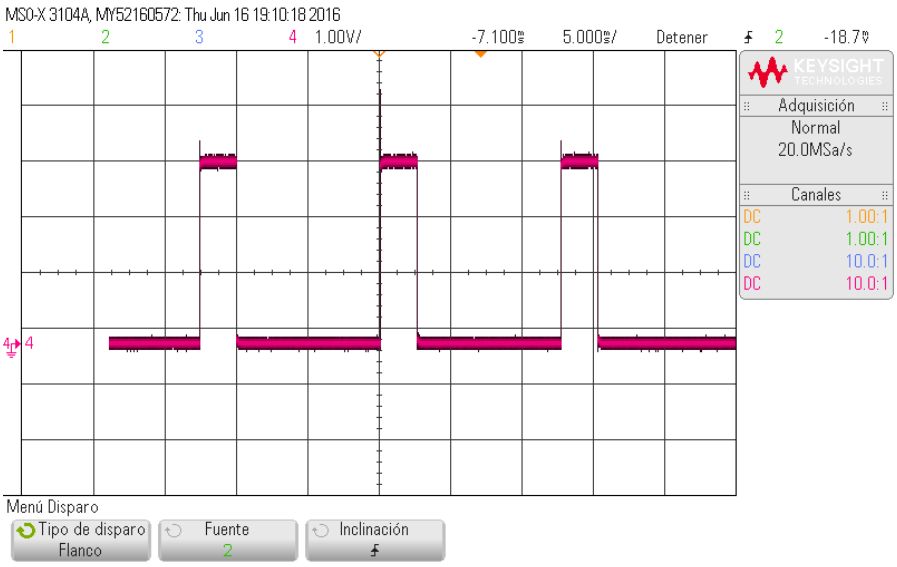

Figura 5-3: Señal PWM 2

En la figura 5-2 observamos que la señal tiene la forma esperada.

$ sudo python /var/www/pwm/cambiarPWM.py '11' '50' '80'

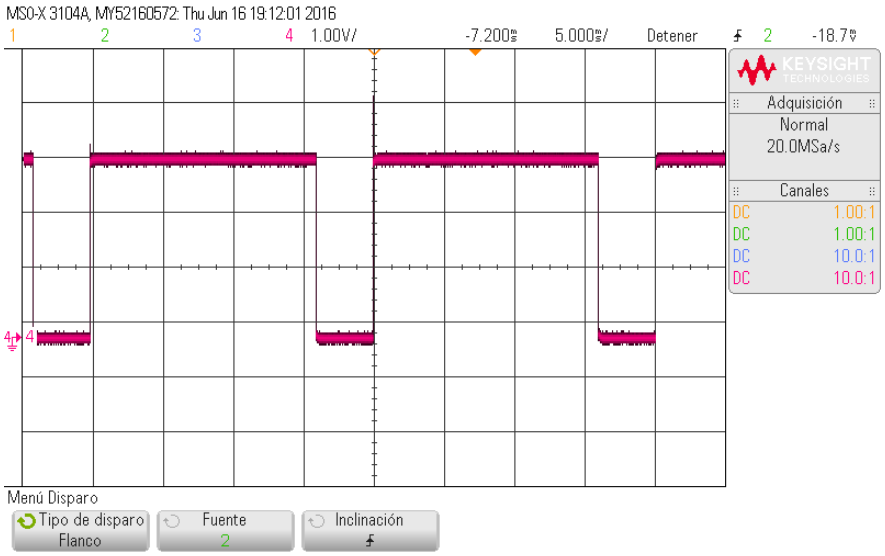

Figura 5-4: Señal PWM 3

En la figura 5-3 observamos que la señal tiene la forma esperada.

Como se puede comprobar, todos los resultados han sido correctos.

Por último, realizamos una llamada al fichero de PararPWM.py para comprobar que el LED se apaga:

$ sudo python /var/www/pwm/pararPWM.py '11'

El LED se apaga correctamente.

5.1.3 Módulo I2C

Se conecta la Rasbperry al módulo expansor con el circuito descrito en la figura C-5 del anexo C.

Realizamos una primera llamada a nuestro fichero EscribirI2C.py para configurar los registros y que se permita la comunicación entre nuestra placa y el dispositivo:

$ sudo python /var/www/i2c/EscribirRegistro.py '0x20' '0x06' '0x00'

Realizamos una llamada a nuestro programa de escritura con los siguientes valores:

- Dirección del dispositivo: 0x20.
- Dirección del registro: 0x02.
- Valor en hexadecimal: 0x11.

$ sudo python /var/www/i2c/EscribirRegistro.py '0x20' '0x02' '0x11'

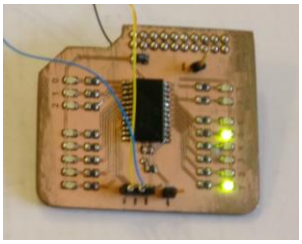

Figura 5-5: Resultado escribir en registro I2C

Se encienden los LEDs correspondientes a los bits con valor de '1' lógico, representando el valor enviado.

Realizamos una lectura del valor del registro 0x02 previamente modificado utilizando el fichero de lectura de registro:

$ sudo python /var/www/i2c/LeerRegistro.py '0x20' '0x02'

Figura 5-6: Resultado lectura I2C

El resultado de la ejecución es '17', valor en decimal de lo que tiene el registro, y '5' en una segunda llamada, lo que nos muestra el correcto funcionamiento la librería.

5.2 Pruebas de Integración

En esta parte de las pruebas se utilizará la página web implementada para incluir todas las llamadas a los ficheros python y comprobar que pueden ejecutarse remotamente uno a uno, analizando los resultados obtenidos y comparándolos con los esperados.

5.2.1 Integración mediante la web

Se incluye en nuestra página web las llamadas a los ficheros de comunicación mediante GPIO, en sus correspondientes botones, y se prueba a realizar las mismas pruebas que en el módulo de prueba unitario pero realizando ahora esta comunicación mediante el uso de los formularios de la web. El resultado es correcto.

A continuación, se integran las llamadas a los fichero de comunicación PWM en sus respectivos botones, y se repite la secuencia de pruebas unitarias del módulo GPIO y seguido las de PWM, comprobando que la integración ha sido un éxito. El resultado es correcto.

Por último se repite el paso anterior pero integrando ahora el módulo de comunicación I2C y llevando a cabo todas las pruebas unitarias de todos los módulos utilizando la interfaz web. Se verifica el correcto funcionamiento de la integración.

6 Conclusiones y trabajo futuro

6.1 Conclusiones

El objetivo principal del trabajo realizado ha sido crear una plataforma de control remoto de bajo coste y generalizada, que integrara su propio servidor web, y proporcionar una serie de librerías que implementasen distintos tipos de comunicación entre esta y los posibles dispositivos a manejar.

Después se eligió el sistema operativo para la plataforma Raspberry Pi: Raspbian, el cual nos proporcionaba la libertad necesaria para llevar a cabo nuestras implementaciones, así como la elección del servidor web Apache para integrar la comunicación del cliente con nuestro hardware.

Una vez instalado todo y puesto en marcha el servidor web, se escogió Python como tecnología de desarrollo para las librerías de control remoto de las aproximaciones GPIO, PWM e I2C, llevando a cabo una implementación de distintos ficheros que estructuran en módulos las distintas funcionalidades necesarias para abarcar todas las posibles acciones sobre dispositivos electrónicos conectados a la plataforma.

Se escogió HTML y PHP como lenguajes para la implementación de la parte de implementación web para dotar de una interfaz sencilla y que permitiera corroborar que las librerías de comunicación habían sido correctamente implementadas, llevando a cabo una implementación básica de lo que necesitaría un programador web que quisiera utilizar las librerías en un futuro.

Con todo ello, se dispone de una plataforma de control remoto ampliable a toda una serie de dispositivos que queramos conectar, valiéndonos de la plantilla web y del número de llamadas que deseemos a nuestras librerías, proporcionándoles el número de puerto y los valores necesarios para las distintas comunicaciones.

6.2 Trabajo futuro

Como propuestas de futuro, se podría llevar a cabo el diseño de una arquitectura maestro – esclavo entre placas Raspberry, para lograr así un mayor número de puertos disponibles para comunicar con otro mayor número de dispositivos.

También se puede estudiar la implementación de otros tipos de comunicación entre la plataforma y los dispositivos como pudiera ser SPI, la cual integra nuestra plataforma y podría incorporar los dispositivos que utilicen dicha comunicación.

Otra posibilidad de trabajo futuro sería la implementación de una aplicación móvil para acceder a nuestro servidor y manejar los dispositivos, proporcionando así una interfaz específica para este tipo de dispositivo de acceso a la red.

Referencias

Arnal, G. (2015). Mercedes me, control remoto del coche desde el móvil. Retrieved from http://www.caranddriverthef1.com/coches/planeta-motor/116858-mercedes-me-control-remoto-del-coche-desde-el-movil

Azzam, N. (2016). Definitions and Differences Between I2C, ACCESS.bus and SMBus. Retrieved from http://ww1.microchip.com/downloads/en/AppNotes/an617.pdf

Bluetooth SIG, I. (2016). Bluetooth technology basics. Retrieved from https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth-technology-basics

Brezo, M. P. (2015). 6 buenos motivos para trabajar con PHP. Retrieved from https://www.lancetalent.com/blog/6-buenos-motivos-para-trabajar-con-php/

CEDOM. (2015). Qué es Domótica. Retrieved from http://www.cedom.es/sobre-domotica/que-es-domotica

Cisco. (2016). WLAN. Retrieved from http://www.cisco.com/c/es_es/solutions/mobility/wlan.html

Debian. (2009). Acerca de Debian. Retrieved from https://www.debian.org/index.es.html

Figueroa, V. (2016). *Características de PHP*. Retrieved from https://es.scribd.com/doc/50288837/Caracteristicas-de-PHP

Gus. (2015). Beaglebone Vs Raspberry Pi 2: Choosing The Right Board. Retrieved from https://pimylifeup.com/beaglebone-vs-raspberry-pi/

I2C-Bus. (2016). I2C - What is that? Retrieved from http://www.i2c-bus.org/i2c-bus/

Infrared Data Association. (2011). IrDA The Secure Wireless Link. Retrieved from http://www.irda.org/

Lansford, J. (1999). HomeRF: Bringing Wireless Connectivity Home. Retrieved from http://www.ieee802.org/11/Tutorial/90548S-WPAN-HomeRF-Tutorial.pdf

María, M. G., & Moreno, J. (2012). ZIGBEE. Retrieved from http://sx-de-tx.wikispaces.com/ZIGBEE

Mercedes-Benz. (2016). Mercedes me connect. Retrieved from https://www.mercedes-benz.com/es/mercedes-me/asi-es-como-funciona-mercedes-connect/

Peralta, G. (2012). Ventajas para los que se ponen la camiseta de python. Retrieved from http://www.maestrosdelweb.com/ventajas-python/

Python Software Foundation. (2016a). About Python. Retrieved from https://www.python.org/about/

Python Software Foundation. (2016b). RPi.GPIO 0.6.2. Retrieved from https://pypi.python.org/pypi/RPi.GPIO

Raspberry PI Foundation. (2012a). GPIO: Raspberry Pi models A and B. Retrieved from https://www.raspberrypi.org/documentation/usage/gpio/

Raspberry PI Foundation. (2012b). What is a Raspberry Pi. Retrieved from https://www.raspberrypi.org/help/what-is-a-raspberry-pi/

Santamaría Velázquez, J. (2015). *Control remoto usando Bluetooth*. Universidad de Sevilla. Retrieved from http://bibing.us.es/proyectos/abreproy/90427/fichero/Control+remoto+usando+Blueto oth.pdf

Siemens AG. (2016). Industrial Remote Communication. Retrieved from http://w3.siemens.com/mcms/automation/es/industrial-communications/industrial-remote-communication/pages/industrial-remote-communication.aspx

The Apache Software Foundation. (2007). What is the Apache HTTP Server Project?

Retrieved from http://httpd.apache.org/ABOUT_APACHE.html
The PHP Group. (2008). ¿Qué es PHP? Retrieved from http://php.net/manual/es/intro-whatis.php
W3C. (2014). HTML & CSS. Retrieved from https://www.w3.org/standards/webdesign/htmlcss
Wikipedia. (2015). GPIO. Retrieved from https://es.wikipedia.org/wiki/GPIO
Wikipedia. (2016). Modulación por ancho de pulsos. Retrieved from https://es.wikipedia.org/wiki/Modulaci%C3%B3n_por_ancho_de_pulsos
ZigBee Alliance. (2015). What is ZigBee? Retrieved from http://www.zigbee.org/what-is-zigbee/

Glosario

IRDA	Infra-Red Data Association
SDA	Serial Data
SCL	Serial Clock
GPIO	General Purpose Input/Output
PWM	Pulse-Width Modulation
LED	Light-Emitting Diode
SMBUS	System Management Bus
GND	Ground
PCB	Printed Circuit Board
UFM	Ultra Fast Mode
SMBUS	Bus de Administración del Sistema
WLAN	Wireless Local Area Network

Anexos

A Manual de instalación

En este manual se explicarán los pasos a seguir una vez obtenida la plataforma Raspberry Pi:

En primer lugar, debemos instalar el sistema operativo deseado, en este caso Raspbian, en la tarjeta SD que utilizaremos a modo de memoria de nuestra plataforma.

Para ello descargamos una imagen del sistema operativo y la instalamos en la tarjeta.

En segundo lugar, instalamos Apache mediante la ejecución de los siguientes comandos dentro de la terminal de nuestra Raspberry:

$ sudo addgroup www-data

$ sudo usermod -a -G www-data www-data

$ sudo apt-get update

$ sudo apt-get install apache2 php5 libapache2-mod-php5

$ sudo /etc/init.d/apache2 restart

Con esto tendremos Apache y PHP instalados en nuestra Raspberry.

A continuación instalamos MySQL y PHPMyAdmin:

$ sudo ifup lo

$ sudo apt-get install mysql-server mysql-client php5-mysql phpmyadmin

Una vez instalado, nos aparecerá una pantalla de configuración preguntando la contraseña que queremos usar y después debemos añadir en el archivo php.ini ($ sudo nano /etc/php5/apache2/php.ini) la siguiente línea antes de la línea “Dynamics Extensions”:

extension=mysql.so

Tras estos escribimos los siguientes comandos:

$ sudo ln -s /etc/phpmyadmin/apache.conf /etc/apache2/conf.d/phpmyadmin.conf

$ sudo /etc/init.d/apache2 reload

Por último, añadimos al final del archivo /etc/apache2/apache2.conf la instrucción Include /etc/phpmyadmin/apache.conf y luego:

$ /etc/init.d/apache2 restart

Para instalar Python en la Raspberry debemos ejecutar:

$ sudo apt-get install python-dev

$ sudo apt-get install python-rpi.gpio

$ sudo apt-get update && sudo apt-get upgrade

B Manual del programador

En este manual se darán ejemplos de llamadas a los ficheros Python implementados para el manejo de los distintos protocolos de comunicación:

Fichero de activación de GPIO: la llamada necesita como argumento un entero que identifica el número de GPIO en numeración BCM que queremos activar. Por ejemplo:

$ sudo python /var/www/leds/gpio/enciende.py '4'

Fichero de desactivación de GPIO: la llamada necesita como argumento un entero que identifica el número de GPIO en numeración BCM que queremos desactivar. Por ejemplo:

$ sudo python /var/www/leds/gpio/apaga.py '4'

Fichero de parpadeo de GPIO: la llamada necesita como argumentos un entero que identifica el número de GPIO en numeración BCM que queremos utilizar; y un entero que representa el tiempo en segundos del parpadeo. Por ejemplo:

$ sudo python /var/www/leds/gpio/parpadea.py '4' '5'

Fichero de activación de PWM: la llamada necesita como argumento un entero que identifica el número de GPIO en numeración BCM que queremos utilizar; un entero que indica la frecuencia en Hertzios a la que queremos que funcione; y un entero entre 0 y 100 que indica el ciclo de trabajo deseado. Por ejemplo:

$ sudo python /var/www/pwm/iniciarPWM.py '11' '100' '50'

Fichero de cambio de valores del PWM: la llamada necesita como argumentos un entero que identifica el número de GPIO en numeración BCM que queremos utilizar; un entero que indica la frecuencia en Hertzios a la que queremos que funcione; y un entero entre 0 y 100 que indica el ciclo de trabajo deseado. Por ejemplo:

$ sudo python /var/www/pwm/cambiarPWM.py '11' '80' '20'

Fichero de desactivación del PWM: la llamada necesita como argumento un entero que identifica el número de GPIO en numeración BCM que queremos apagar. Por ejemplo:

$ sudo python /var/www/pwm/pararPWM.py '11'

Fichero de escritura por I2C: la llamada necesita como argumentos un número hexadecimal de dos cifras que indica la dirección del dispositivo conectado; un número hexadecimal de dos cifras que indica la dirección del registro en el que queremos escribir; un número hexadecimal de dos cifras que representa el valor que queremos enviar. Por ejemplo:

$ sudo python /var/www/i2c/EscribirRegistro.py '0x20' '0x02' '0x11'

Fichero de lectura por I2C: la llamada necesita como argumentos un número hexadecimal de dos cifras que indica la dirección del dispositivo conectado; un número hexadecimal de dos cifras que indica la dirección del registro en el que queremos escribir. Por ejemplo:

$ sudo python /var/www/i2c/LeerRegistro.py '0x20' '0x02'

C Montaje de experimentos

Circuito LED – Plataforma (Diseño):

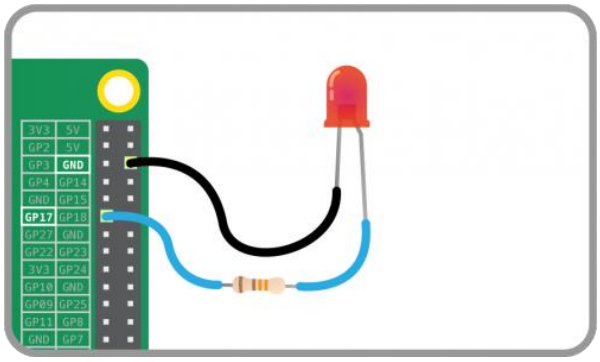

Figura C-1: Circuito GPIO-LED

El circuito real se muestra en la figura C-2.

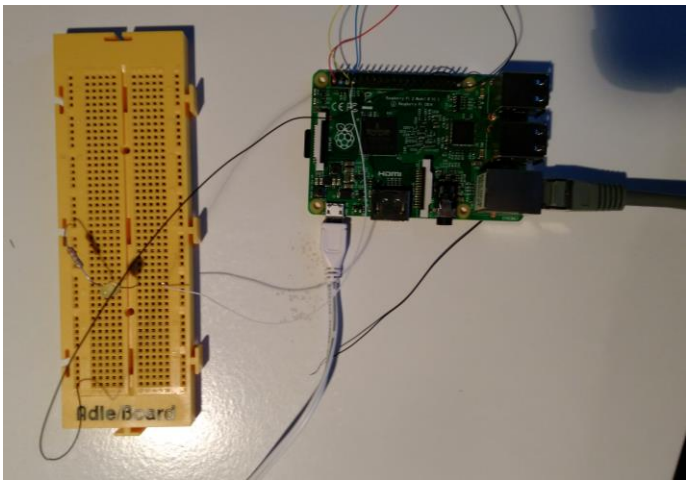

Figura C-2: Circuito real GPIO-LED

En la comunicación I2C entre un dispositivo expansor y nuestra plataforma, utilizamos un circuito como este:

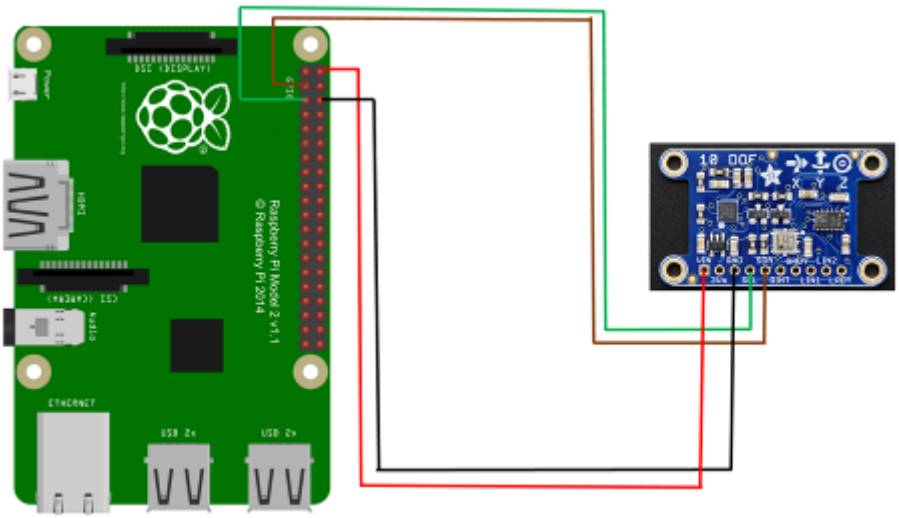

Figura C-3: Circuito I2C-placa expansora

La conexión real entre la Raspberry Pi y la placa expansora se muestra en la figura C-4.

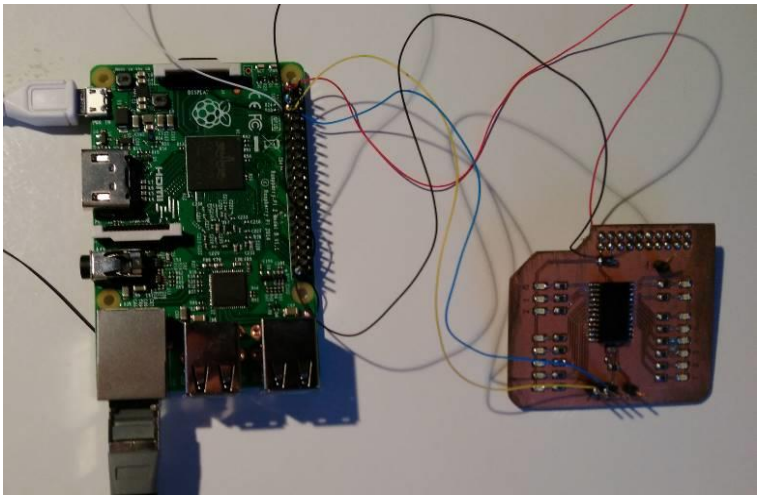

Figura C-4: Circuito Raspberry - Placa expansora

Como dispositivo básico a controlar mediante este I2C disponemos de un disco duro antiguo perteneciente a la universidad (figura C-5).

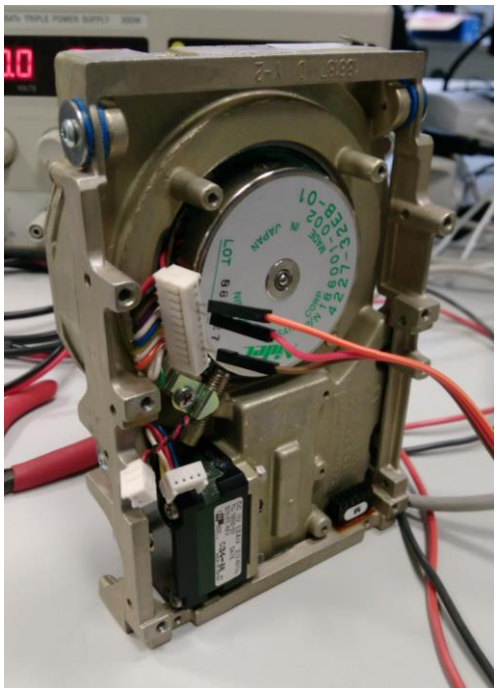

Figura C-5: Disco duro

La vista de la página web que muestra la funcionalidad implementada es la siguiente:

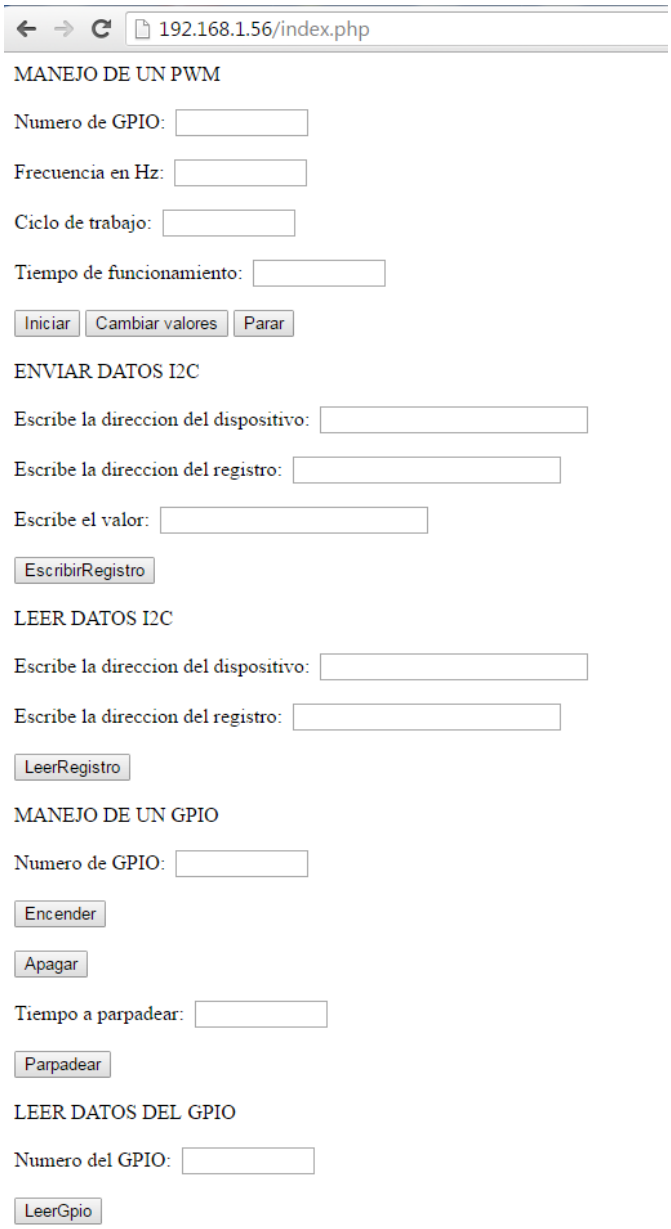

Figura C-6: Vista página web

La figura C-7 muestra una fotografía real del PCB diseñado para controlar el motor del disco duro:

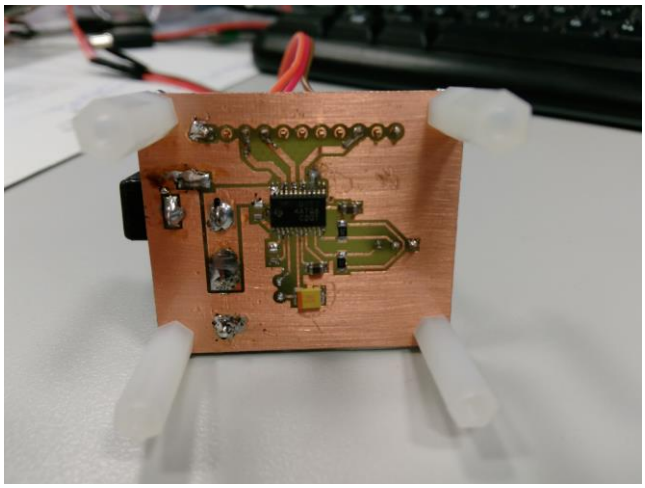

Figura C-7: PCB motor disco duro